DOCUMENT RESUME

ED 060 621 EM 009 627

AUTHOR	Bork, Alfred M.; And Others
TITLE	Teaching Conversations with the XDS Sigma 7.
INSTITUTION	California Univ., Irvine. Physics Computer Development Project.
SPONS AGENCY	National Science Foundation, Washington, D.C.
PUB DATE	7 Dec 71
NOTE	28p.
EDRS PRICE	MF-$0.65 HC-$3.29
DESCRIPTORS	Computer Assisted Instruction; *Computer Graphics; *Display Systems; Guides; *Manuals; Programing Languages; Science Instruction
IDENTIFIERS	*Sigma 7 Computer

ABSTRACT

A manual describes the use of graphic commands in student-computer dialogues. How to construct axes, windows, boxes and various other computer displays is explained, in particular reference to the ARDS 100 and TEKTRONIX 4002 and 4010 computer terminals. Concrete examples of displays are included. The appendix contains an explanation of the use of numbers and arrays. (RB)

ED 060621

TEACHING CONVERSATIONS WITH THE XDS SIGMA 7

GRAPHIC DIALOG FACILITIES

Alfred Bork
Estelle Warner
John Collins

December 7, 1971

EM 009627

CONTENTS

CONTENTS

INTRODUCTION

With the availability of inexpensive graphic terminals, a new dimension opens up in the use of computers for interactive teaching. The teacher has the ability to provide the student with endless and imaginative illustrations for computer conversations and he can invite the student to experiment graphically with the effects of changing parameters or equations or changing the framework of the experiment.

Working with ARDS 100 and Tektronix 4002 and 4010 terminals, our dialog facilities have been extended with graphic commands which allow the author to use the terminal facilities in a simple manner. (The coding can easily be modified to work for other graphic terminals.) This new manual* is concerned with the use of these added instructions.

*There are three previous manuals in the series (Appendix II).

pensive graphic terminals, a new dimen-
omputers for interactive t aching.
o provide the student with endless and
computer conversations and he can
ent graphically with the effects of
ons or changing the framework of the

tronix 4002 and 4010 terminals, our
xtended with graphic commands which
erminal facilities in a simple manner.
ified to work for other graphic ter-
concerned with the use of these added

nuals in the series (Appendix II).

CHAPTER 1
INITIALIZING THE DIALOG

The first command in a dialog is, as usual,

```
        SYSTEM     DIALOG
```

This must be followed by NAME (or START), and then DEVICE (see below). END DIALOG is, as before, the last command in the program.

If the dialog is divided into sections assembled separately, the DEVICE command should appear only in the first segment to be executed.

CHAPTER 2
PROCEDURES FOR DETERMINING THE TERMINAL

DEVICE allows the student to indicate which terminal is in use, and sets a flag internally.

```
        DEVICE     ARDS,TEK
```

or

```
        DEVICE     ARDS,TEK,'TEK 4010'
```

asks the student which terminal he is using. If he responds with one of the arguments, the program proceeds; otherwise it tells him on which terminals the program can run, and gives him the option of stopping if he is on some other kind. 'TEK' implies a 4002 terminal.

DEVICE must follow START or NAME and must precede any graphic command.

CHAPTER 3

CONTROLLING THE SCREEN

A. ERASE

ERASE will erase the screen. On the ARDS and TEK 4010 (not Tek 4002), it will also reset the cursor, the current beam position, to the top left corner.

B. HOME

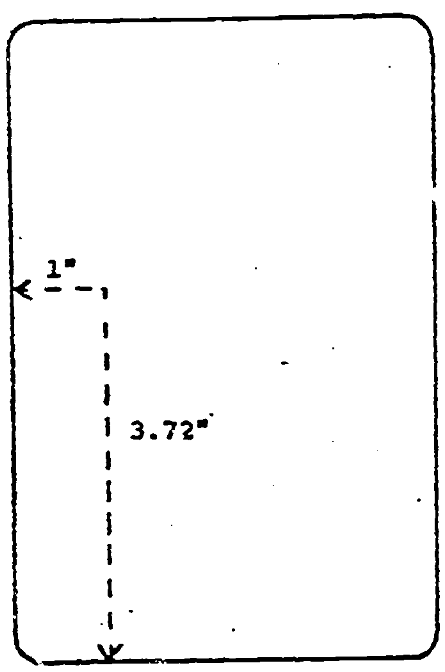
HOME moves the cursor to the top left of the screen.

C. SETPOINT

SETPOINT moves the cursor to the indicated point on the screen. It is useful for resetting the cursor to the point where printing of dialog material is to be resumed, after plotting is finished. It can be called in several ways. The most basic,

```
SETPOINT      (FS'1',FS'3.72')
```

will position the cursor one inch over and 3.72 inches up from the lower left-hand corner of the screen on either terminal.



SETPOINT (FS'1',FS'3.72')

Fig. 1

RDS and TEK 4010 (not Tek
the current beam position,

of the screen.

ated point on the screen. It
the point where printing of
er plotting is finished. It
ost basic,

)

and 3.72 inches up from the
on either terminal.

SETPOINT (FS'1',FS'3.72')

Fig. 1

If A is a symbolic reference to a floating short '1' (all numeric arguments* are floating short unless otherwise specified), and B is a symbolic reference to 3.72, then

```
        SETPOINT     (A,B)
```

will have the same result.

```
        SETPOINT     (FS'1',B)
```

is also valid. SETPOINT can be called with the terminal specified:

```
        SETPOINT     (ARDS,(A,B)),(TEK,(B,A))
```

will position the cursor at (1,3.72) on an ARDS, and (3.72,1) on a TEK.

SETPOINT is independent of all scaling and windowing data, i.e., it sets absolute points.

D. SCREEN

SCREEN is a comprehensive command, with the options ERASE, HOME, SETPOINT, each used as above.

```
        SCREEN     ERASE,(SETPOINT,(A,B))
```

will erase the screen and position the cursor at the point whose coordinates (in inches) are given in A and B.

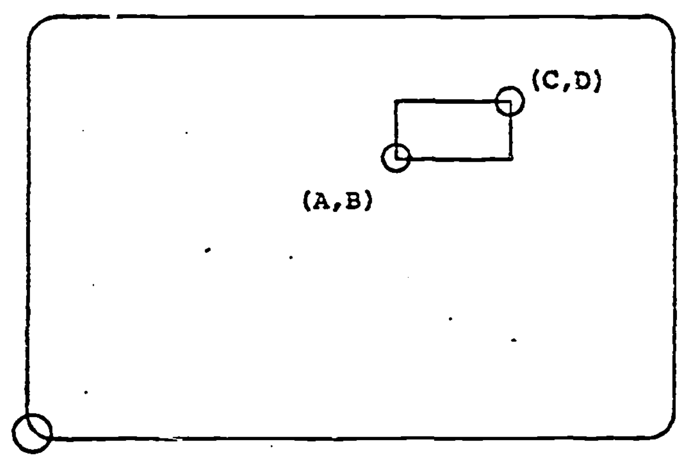
*Cp Appendix I.

CHAPTER 4
WINDOWING

Windowing is the setting up of a separate rectangular area on the screen in which plotting is to take place. It enables the author to display a plot on one part of the screen, while continuing the dialog on another, or to show several plots on the screen at the same time.

A. WINDOW

WINDOW defines the rectangular space in which a curve or set of curves is to be displayed.



WINDOW (A,B),(C,D),BOX

Fig. 2

specifies a window with lower left corner displaced A, B inches and upper right corner displaced C, D inches, measured from the lower left corner of the screen. A, B, C, D are, as in the SETPOINT command, floating short numbers or variables whose values are floating short numbers. The window is not drawn unless BOX is specified.

B. BOX

BOX draws

an option

or

do the sa

A single

terminal,

account:

A WINDOW

More than

the scree

refer to

no window

The comma

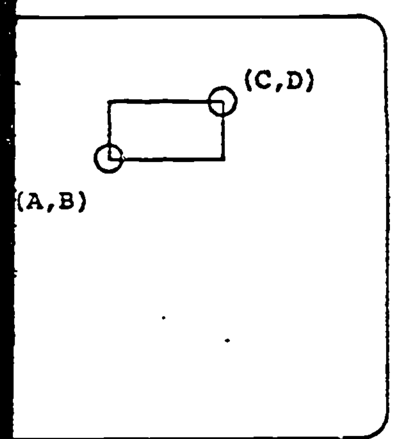
should be

full scre

The defau

a separate rectangular area on the
take place. It enables the author to
he screen, while continuing the dialog
plots on the screen at the same time.

space in which a curve or set of curves



A,B),(C,D),BOX

ig. 2

eft corner displaced A, B inches and
, D inches, measured from the lower left
, D are, as in the SETPOINT command,
ables whose values are floating short
awn unless BOX is specified.

B. BOX

BOX draws the last window specified on the screen. It can be used as an option with WINDOW or as a separate command.

```
WINDOW     (A,B),(C,D),BOX
```

or

```
WINDOW     (A,B),(C,D)
BOX
```

do the same thing.

A single window command may specify the window differently for each terminal, so that differences in screen shape can be taken into account:

```
WINDOW     (ARDS,(A,B),(C,D)),(TEK,(A,C),(K,S))[,BOX]
```

A WINDOW command releases any previous window or scaling information. More than one window (containing curves, axes, etc.) can be shown on the screen at one time, but any CURVE, AXES, or SCALE command will refer to the most recently defined window (or to the full screen if no window is in effect.)

The command

```
NOWINDOW
```

should be used when the author wants to return to plotting on the full screen; it also releases any previous scaling information. The default is NOWINDOW.

CHAPTER 5
SCALING

The programmer can either allow the set of data to determine its own scale by specifying the MAX option when he calls CURVE or AXES (see below), or he can use the SCALE command separately, or the SCALE option with the command CURVE or AXES.

The scaling will take effect within whatever window, if any, is currently defined. If the full screen is in effect its full width will be used for the horizontal range defined by the MIN-MAX information, and the full height for the specified vertical range. If WINDOW is in effect, the full window space will be utilized.

SCALE enables the author to choose the scale in his own coordinates, related to the values in use. It is useful for plotting a series of different curves, not all containing the maximum and minimum, on the same frame. A typical procedure call for a scale is

```
SCALE    (XMIN,XMAX),(YMIN,YMAX)[,(ZMIN,ZMAX)]
```

where the arguments are floating short numbers* or variables whose values are floating short numbers. (If they are variables, they must be defined before they are used in a graphic command.)

For a three dimensional curve (projected into two dimensions) the third argument is included.

Any following WINDOW or NOWINDOW command wipes out the scaling information as well as substituting a new window size, (or restoring the full screen use). A second SCALE command (or the use of the MAX option or the SCALE option with the command CURVE or AXES) replaces the old scaling information with the new, but does not affect the current WINDOW (or NOWINDOW) state.

---

*Appendix I.

he set of data to determine its own
n when he calls CURVE or AXES (see
ommand separately, or the SCALE
AXES.

in whatever window, if any, is
screen is in effect, its full width
range defined by the MIN-MAX informa-
e specified vertical range. If
ndow space will be utilized.

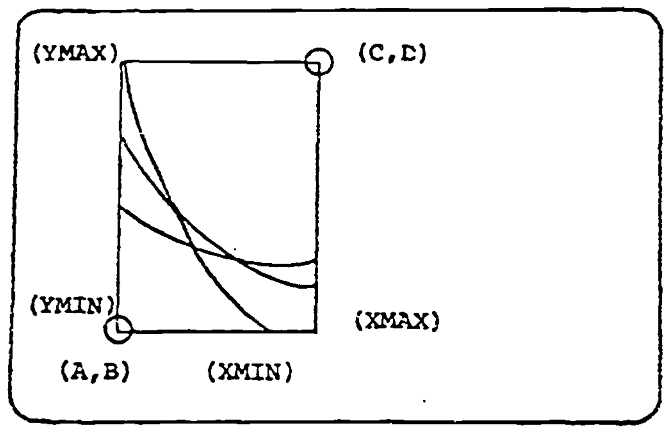
se the scale in his own coordinates,
t is useful for plotting a series of
ning the maximum and minimum, on the
call for a sc is

(YMIN,YMAX)[,(ZMIN,ZMAX)]

short numbers* or variables whose
s. (If they are variables, they
used in a graphic command.)

rojected into two dimensions) the

command wipes out the scaling
ing a new window size, (or restoring
SCALE command (or the use of the
ith the command CURVE or AXES) replaces
the new, but does not affect the
te.



WINDOW	(A,B),(C,D),BOX
SCALE	(XMIN,XMAX),(YMIN,YMAX)
CURVE	(X1,Y1,N1)
CURVE	(X2,Y2,N2)
CURVE	(X3,Y3,N3)

Fig. 3

CHAPTER 6

PLOTTING

The dialog graphic facilities permit both curve plotting and point plotting. These facilities are implemented with the CURVE and POINT procedures.

A. CURVE

CURVE plots arrays of data* in 2 or 3 (projected) dimensions (data storage must be defined by the programmer before the command is used). The CURVE command has numerous options, permitting a variety of plotting modes. The basic procedure call has the form:

```
CURVE     (V1,V2,N)
```

In this case V1 and V2 are names of arrays containing at least N values. N is the number of values in the array to be used in plotting, a variable or an integer. Without altering the values in the arrays, they are scaled and windowed according to the data specified by the last SCALE and WINDOW (or NOWINDOW) commands, or the last AXES command. If there is no previous scaling information, the 'MAX' option (see below) is issued, and the points are connected with straight lines output to the terminal.

If the option SCALE is used within the CURVE command, as in

```
CURVE     (X,Y,N),(SCALE,(HMIN,HMAX),(VMIN,VMAX))
```

the new scale information is stored, and N points of the X and Y arrays are plotted according to the window and scale data now defined.

```
CURVE     (MAX,(X,Y,N))
```

*It is often convenient to compute these arrays in FORTRAN subroutines. The user is also referred to the section (Appendix I) on the procedures DEFARRAY and STORARRAY.

permit both curve plotting and point
implemented with the CURVE and

2 or 3 (projected) dimensions (data
programmer before the command is used).
options, permitting a variety of
edure call has the form:

es of arrays containing at least N
lues in the array to be used in plot-
. Without altering the values in the
ndowed according to the data specified
or NOWINDOW) commands, or the last AXES
ous scaling information, the 'MAX'
and the points are connected with
erminal.

thin the CURVE command, as in

CALE,(HMIN,HMAX),(VMIN,VMAX))

tored, and N points of the X and Y
o the window and scale data now defined.

N))

pute these arrays in FORTRAN sub-
ferred to the section (Appendix I)
STORARRAY.

will find the maximum and minimum of the X and Y arrays and store this as the new scale data. Then the array points will be plotted with connecting lines in the currently defined window.

To plot (in two dimensions) a three dimensional set of points, the call could be:

```
CURVE     (X1,X2,X3,K)
```

where K is the number of points. The same options can be used as with plots for two variables.

You may begin plotting at any position in the arrays by

```
CURVE     ((X,5),(Y,12),30)
```

will plot the figure described by the 30 pairs of coordinates, beginning with the fifth element of X and the twelfth element of Y.

Any of the vector arguments to CURVE may be indirect addresses.

The option CENTER operates like MAX, but in addition will cause the graph to appear with its (0,0) or (0,0,0) point in the center of the screen or window:

```
CURVE     (CENTER,(X,Y,N))
```

Other options are possible: the curve can be dotted (instead of solid, which is the default). This is called by

```
CURVE     (A,ANEW,12),DOT
```

It is also possible to save the code generated by a CURVE command and then call for a reshowing of that curve:

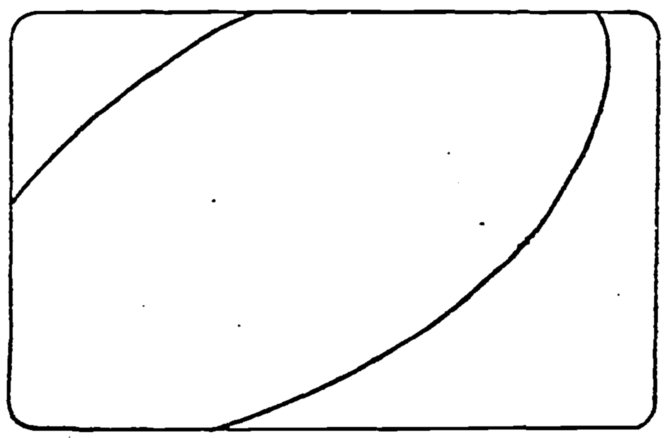
```
CURVE     (X,Y,Z,40),(SAVE,XX)
```

stores the code, (both command and data characters), for the plot in an array XX (storage provided by programmer); and the command

```
CURVE      (AGAIN,XX)
```

draws it unchanged without having to recalculate. If a point specified by the arrays in CURVE should fall outside the window (or full screen, if no window has been specified), a line will be drawn to the intersection point with the boundary, and the CURVE will be discontinued.

If the option REENTRY is specified with CURVE, the first following point which falls inside the window will be plotted, with a segment of the line to the preceding outside point and plotting will be resumed.



CURVE   (X,Y,120),(REENTRY)

Fig. 4

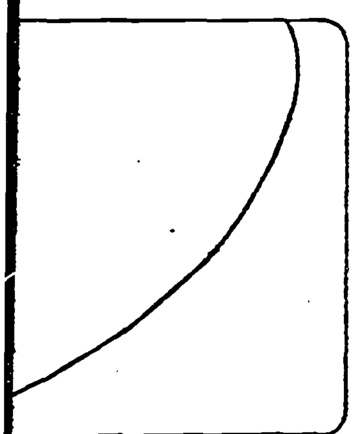
B. POINT

This command enables the author to plot discrete points rather than a continuous curve. It functions like CURVE, and all options used with CURVE are valid with POINT (except DOT!).

d data characters), for the plot in an
grammer); and the command

to recalculate. If a point specified
d outside the window (or full screen,
a line will be drawn to the inter-
and the CURVE will be discontinued.

d with CURVE, the first following
ow will be plotted, with a segment
ide point and plotting will be resumed.



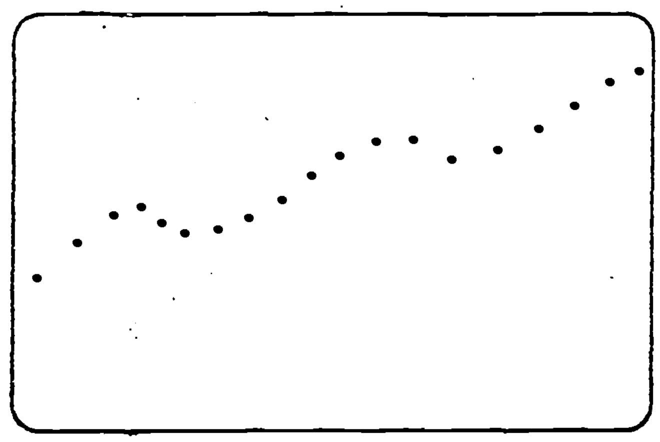
120),(REENTRY)

4

to plot discrete points rather than
s like CURVE, and all options used
(except DOT!).

POINT (X,Y,N)

will plot N points of the arrays X, Y, (Fig. 5) using previously defined SCALE factors.



POINT (X,Y,N)

Fig. 5

POINT (X1,X2,X3,N3)

will produce a two dimensional projection of the N3 points on the X1, X2, X3 curve.

C. LINE.

The command:

LINE (X1,Y1),(X2,Y2)

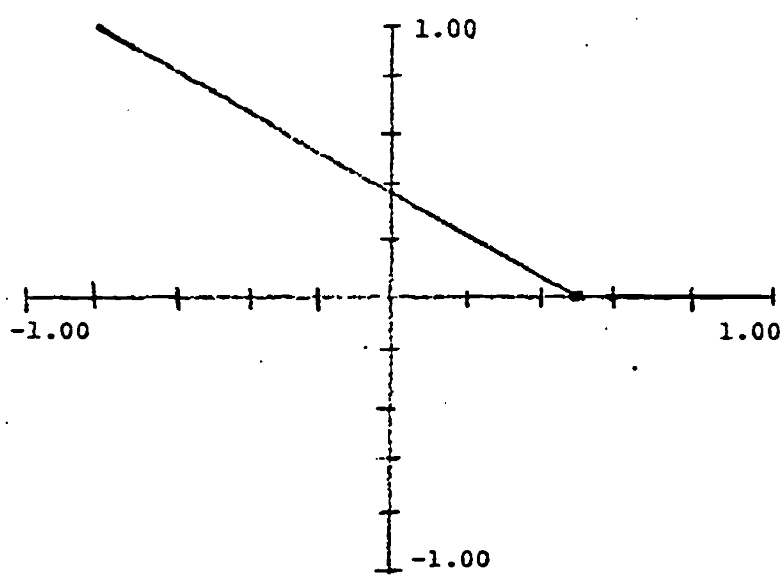
will draw a line from point (X1,Y1) to point (X2,Y2), where X1, Y1, X2, Y2 are floating short numbers, and are within the limits set by SCALE. For example, the program:

```
SCALE     (FS'-1',FS'1'),(FS'-1',FS'1')

AXES      LIMITS

LINE      (FS'.5',FS'0'),(FS'-.8',FS'1')
```

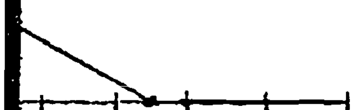
would produce the graph:



1'),(FS'-1',FS'1')

'),(FS'-.8',FS'1')

00



1.00

.00

CHAPTER 7
DRAWING AXES

You can specify axes for both two and three dimensional systems, label axes, and indicate numerical values to determine the scale. The positive part of each axis is a solid line and the negative dotted. Axes are drawn in proper relation to the scaled plot on the full screen or the window, whichever is in effect.

The basic call

```
        AXES
```

will use whatever information is in store for WINDOW and SCALE, and draw horizontal and vertical axes through the 0,0 (or 0,0,0) point. If this point is not within the range represented in the plot, the axes will be drawn through the lower left corner of the window (or screen).

```
        AXES     (DIM,3)
```

will do the same for three dimensions.

Various options can be used:

> Example One
> AXES   (SCALE,(HMIN,HMAX),(VMIN,VMAX))
> will store the new data as scale information and draw the axes accordingly.
>
> Example Two
> AXES   (MAX,(X,Y,N))
> will find the H and V minimum, maximum values of the N points in the X and Y arrays, use these to compute and store scale data, and draw the axes. This can be extended to:

Example Three

```
AXES      (MAX,(X,Y,Z,K))
```

Another useful option for AXES is LABELS.

```
AXES      (MAX,(X,Y,N)),(LABELS,'X','Y')
```

would function as in Example Two above, but would also label the axes. The arguments for LABELS must be strings. It can also be used in three dimensions.

LIMITS will print the maximum and minimum values for each axis at the axis end-points

```
AXES      (MAX,(A,B,50)),LIMITS
```

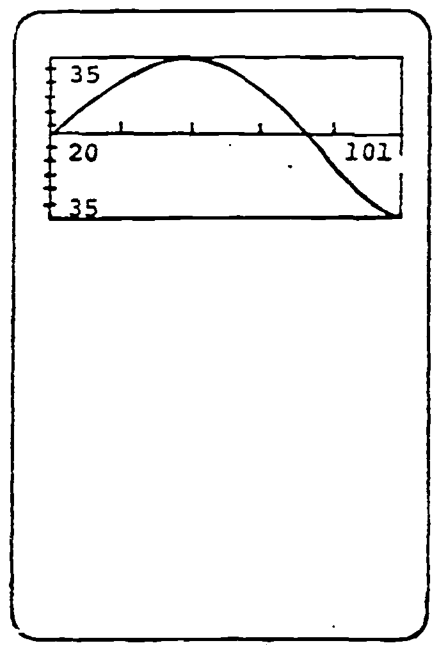
15

'X','Y')

out would also label the
strings. It can also be used

n values for each axis at

16




```
WINDOW    (FS'1.5',FS'4'),(FS'4.5',FS'6.6'),BOX
CURVE     (MAX,(X,Y,50))
AXES      LIMITS
        or
WINDOW    (as above)
AXES      (MAX,(X,Y,50)),LIMITS
CURVE     (X,Y,50)
```

Fig. 6

NOTICS
Will suppress the drawing of tic marks on the axes. (Default = tics).
TICS
Used without arguments,

```
AXES      (DIM,3),TICS
```

Will draw five evenly spaced tic marks on the larger sections of each axis, and appropriately fewer on the shorter. Used with arguments,

```
AXES     (LABELS,'X','Y'),(TICS,FS'.1',FS'10')
```

will draw the two axes, labeled as indicated, and draw tic marks at intervals of .1 on the 'X'-axis and 10. on the 'Y' axis, where .1 and 10. are chosen relative to the data arrays being plotted, and will be windowed and scaled exactly as the data arrays.

```
AXES     (LABELS,'X','Y','Z'),(TICS,FS'.1',FS'10',FS'1.5')
```

will do the same, in three dimensions.

Example:

```
AXES    (CENTER,(XA,YA,N)),(LABELS,'X1','X2'),LIMITS,;
        (TICS,FS'1',FS'.2')
```

might give (in the case of a sine function):

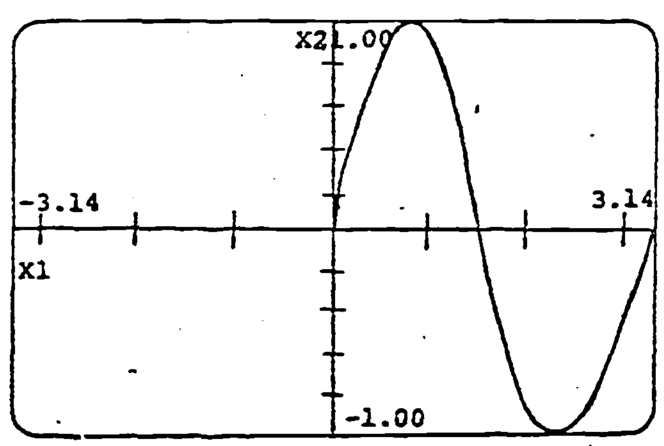


Fig. 7

s on the larger sections of
the shorter. Used with

ICS,FS'.1',FS'10')

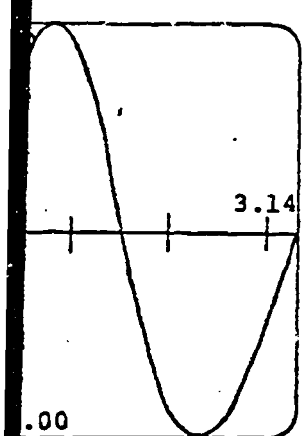
licated, and draw tic marks at
0. on the 'Y' axis, where .1
ta arrays being plotted, and
s the data arrays.

'),(TICS,FS'.1',FS'10',FS'1.5')

.

X1','X2'),LIMITS,;

ction):



CHAPTER 8

The user can store and reshow curves. The graphic data is stored on a disk file.

A. SAVEPLOT

The save command

SAVEPLOT "argument"

will save on disk the last figure displayed, in a file in the account in which the program is being run. This file name is specified in the argument to "SAVEPLOT" and may be a character string literal ("FILE1") or the name of a location containing a character string. The file name should be less than 16 characters. When saved it will be prefixed with the letters "PIC" to help identify these saved plot files in the user's account.

Only the last figure displayed is saved, without axes. For plots using axes, the axes must be drawn previous to the curve. Scaling and windowing information are automatically saved with the plot.

B. SHOWPLOT

The command for restoring a plot is

SHOWPLOT "argument"

where argument is the same as for "SAVEPLOT." It is not necessary to specify the "PIC" prefix when restoring a saved plot. To restore the axes as they were drawn for a particular saved figure, you need only execute an "AXES" command after restoring the saved plot, before making any changes in the scaling or windowing. All "AXES" options are valid.

APPENDIX I

A. Notes on Numbers

1. Constants and Variables. A constant is a quantity whose value is explicitly stated, for example, 3.14159, or 5. A variable is a numerical quantity that is referenced by name rather than by its explicit appearance in a program statement. During the execution of the program, a variable may take on many values. A variable is identified and referenced by a label (if it is a real number or integer), or by a counter name (if it is a counter).

2. Kinds of Numbers. The dialog procedures recognize and use three kinds of numbers: real (floating point), counters, and integers. Each is stored in the computer in different form, and each has its specific uses. These are discussed in the following paragraphs, and Figure 1 (Appendix I) charts dialog commands appropriate to each kind of number.

a. REAL numbers are approximate representations of the range 5.398×10^{-79} to 7.237×10^{75} (in the SIGMA 7). Typical real numbers are 3.2, 100.75, 3E + 10, -.0024, etc. These are the numbers commonly used for computation. They are also called 'floating point' numbers (referring to the form in which they are stored and manipulated in the computer). A real number stored in one machine word is a 'floating short' number.

Those procedures use these single precision real numbers. In the SIGMA 7 this affords a precision of 6+ significant digits.

In most DIALOG commands for real numbers, either the constant (given as, for example, FS'2.3'), or the variable form can be used.

b. COUNTERS are variables with integer values in the range 0 to 255. They are used to control the program flow or to keep count of student scores, the number of trials, etc. They

les. A constant is a quantity whose
ample, 3.14159, or 5. A variable is
enced by name rather than by its
tatement. During the execution of
n many values. A variable is
el (if it is a real number or integer),
counter).

The dialog procedures recognize
real (floating point), counters,
e computer in different form,
These are discussed in the following
I) charts dialog commands appropriate

approximate representations
37 x 10^{75} (in the SIGMA 7). Typical
10, -.0024, etc. These are
putation. They are also called
ng to the form in which they are
puter). A real number stored in
hort' number.

precision real numbers. In the
of 6+ significant digits.

numbers, either the constant (given
variable form can be used.

riables with integer values in the
control the program flow or to
number of trials, etc. They

differ from integers in the way they are stored in the computer, (they are packed four to a word in the SIGMA 7), so they cannot be used interchangeably with integer 9. Two commands exist to convert from counter to integer, and vice versa: CTONUM and NUMTOCT. Other commands are listed in Figure 1, Appendix I.

c. INTEGERS are precise representations of the range of integers from -2^{31} to $+2^{31}-1$. They can be used for indexing arrays, either those used in graphic commands, or those used in calls to FORTRAN routines. Integers are stored one to a word, and cannot be used interchangeably with counters without conversion (cp. CTONUM and NUMTOCT).

The commands DEFNUM and STORENUM can be used to define integer variable and place values in them, BUMPNUM can be used to increment (or decremen an integer variable, and TOINDEX can be used to test on an integer valu Various other DIALOG commands are usable with integers; they are listed in Fig. 1, Appendix I.

B. <u>Array Storage and Indices</u>

Since the graphic routines depend on arrays, we think it desirable to summarize the process of setting up and working with arrays, and indexing them.

The first element in an array always has the index 1. If the index is a variable, it must be defined <u>before</u> it is used in a command. All arrays should also be defined before using.

Arrays can be defined with DEFARRAY (see below), and values can be stored in the array either by STORARRAY or by using the command FORTRAN with the array name(s) as argument(s) to access a Fortran subroutine written to compute values for the array(s). The latter is more generally useful.

An index (integer variable) can be defined by DEFNUM, and a value stored in it by DEFNUM or STORENUM. It can be incremented

(decremented) by BUMPNUM, and a test for branching can be made on it by TOINDEX. Those commands which are not (at present) listed in the SYSTEM USERS MANUAL* are given here. A sample program using some of the arrays and indexing structures are given in Figure 2, Appendix I.

<u>DEFARRAY</u>
reserves space for one or more arrays.

```
DEFARRAY      (VECTOR,100)
```

will set up 100 spaces for array VECTOR. It can have multiple arguments:

```
DEFARRAY      (A,10),(B,20),(C,60)
```

will reserve the indicated space for each of A, B and C. The array size must be an integer constant. The command can be used anywhere in the program, (after the initial statements).

<u>STORARRAY</u>
inserts values into an array defined elsewhere. If Z contains FS'1' and Q contains FS'.07'

```
STORARRAY      VECTOR,(FS'1.02',Z,Q,FS'2)
```

will place the listed values into VECTOR (1), (2), (3), (4). For graphic use the values should be floating short; for general use they can also be integer.

The vector name may be indexed:

```
STORARRAY      (A,5),(R,S,T,FS'2.1')
```

will store the floating short numbers in the list into A, beginning with the fifth element of A.

*See Appendix II

for branching can be made
ich are not (at present) listed
n here. A sample program using
ctures are given in Figure

s.

TOR. It can have multiple argu-

,(C,60)

each of A, B and C. The array
he command can be used anywhere
tatements).

elsewhere. If Z contains FS'1'

1.02',Z,Q,FS'2)

ECTOR (1), (2), (3), (4). For
oating short; for general use they

T,FS'2.1')

rs in the list into A, beginning


```
        STORARRAY     (B,N),(FS'3.2',FS'6.4')
```

will store 3.2 and 6.4 into B(50) and B(51) if N contains the integer 50.

The STORARRAY command must be placed where it will be executed during the program run, whereas the defining commands (DEFARRAY,etc.) need not be executed.

The following commands are useful for indexing with CURVE and STORARRA

<u>DEFNUM</u>
defines (and, optionally, stores) an integer number.

```
        DEFNUM     INDEX,3
```

will define the label INDEX, and place an integer 3 in it.

```
        DEFNUM     INDEX,N
```

is equivalent if N contains the integer 3.

<u>STORENUM</u>
can store an integer or integer variable in a previously defined location:

```
        STORENUM     INX,5
```

<u>BUMPNUM</u>
increments an integer variable by an integer constant:

```
        BUMPNUM     INDEX,-3
```

will add -3 to the integer in INDEX. The second argument may not be a variable name.

<u>TOINDEX</u>
is a conditional transfer, testing on an integer variable.

```
TO INDEX     A1,(ITEST,LE,10)

TO INDEX     A1,(ITEST,LE,K)
```

will each transfer to A1 if the integer in ITEST is LE 10 (or the integer in K). Any of the usual relations: LT, EQ,GT etc. can be used. If omitted,

```
TOINDEX     A1,(ITEST,5)
```

the relation GE is assumed.

ing on an integer variable.

T,LE,10)

T,LE,K)

integer in ITEST is LE 10 (or the
l relations: LT, EQ,GT etc. can be

,5)

FIGURE 1, APPENDIX I

COMMANDS THAT CAN BE USED WITH

Floating Pt. Numbers	Counters	Integers
AROUND, BETWEEN	TO, TOCTR	DEFNUM
DEFNUM	ADDCOUNT	STORENUM
	CTARITH	DEFARRAY
STORENUM	CTOUT	STORARRAY
DEFARRAY	CTWRITE	BUMPNUM
STORARRAY	SWITCH	STACK
DEFTABLE	BUMP(AUGMENT,etc.)	SCAN
STACK	COUNTER	TOINDEX
OUTABLE	RESET	CTONUM
OUTNUM	CTONUM	NUMTOCT
WRITNUM	NUMTOCT	
SCAN		
SCAN#		
NUMBER		
PLOT,GRAPH		
RANDOM		

```
          SYSTEM    DIALOG
          START
          REF       VALUES
          DEVICE    TEK,ARDS
          DEFNUM    ITEST,N
          DEFARRAY  (X,100),(Y,100)
          DEFNUM    INDEX,1
          STORENUM  N,100
          STORENUM  ITEST,99
          FORTRAN   VALUES,(X,Y,(N,1))
          AXES      (MAX,(X,Y,100))
LOOP      CURVE     ((X,INDEX),(Y,INDEX),2)
          BUMPNUM   INDEX,2
          TOINDEX   LOOP,(INDEX,LT,ITEST)
            .
            .
            .
```

(Program continues)

DIALOG LOOP USING INDEXING

This program will plot the line segments connecting every other pair of X,Y coordinates.

FIGURE 2, APPENDIX I

)

DEX),2)

TEST)

inues)

INDEXING

ents connecting every other

NDIX I

APPENDIX II - References

1. Bork, A., "Physics Teaching and Computer Languages." Preprint.
2. Bork, A., Notions about Motion, (W. H. Freeman & Company San Francisco); Preliminary edition - 1970.
3. Bork, A., "Computer-Based Mechanics," in the Proceedings of the COMUSE Conference, Illinois Institute of Technology, August, 1970.
4. Bork, A., FORTRAN for Physics, (Addison-Wesley, 1966).
5. Bork, A., Luehrmann, A., and Robson, J., Introductory Computer-Based Mechanics, (Commission on College Physics, 1968).
6. Bork, A., "The Computer in a Responsive Learning Environment - Let a Thousand Flowers Bloom." Dartmouth Conference on Computers in Education, 1971.
7. Bork, A. and Mosmann, C., "Teaching Conversations with the XDS Sigma 7 - System Description." Preprint.
8. Mosmann, C. and Bork, A., "Teaching Conversations with the XDS Sigma 7 - System Users Manual." Preprint.
9. Warner, E., and Bork, A., "Teaching Conversations with the XDS Sigma 7 - System Maintenance Manual." Preprint.
10. Bork, A. and Sherman, N., "A Computer-Based Dialog for Deriving Energy Conservation for Motion in One Dimension," American Journal of Physics.
11. Bork, A., "Advice to Dialog Writers." Preprint.
12. Bork, A. and Robson, J., "A Computer Simulation for the Study of Waves." Preprint.
13. Bork, A., "Inexpensive Timeshared Graphics on the Sigma 7." Preprint.
14. Bork, A. and Peckham, H., "Computer Needs for Teaching Physics," Preprint, 1970.